대 한 민 국 특 허 청
KOREAN INDUSTRIAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Industrial Property Office.

출 원 번 호 : 특허출원 2001년 제 6901 호
Application Number

출 원 년 월 일 : 2001년 02월 13일
Date of Application

출 원 인 : 삼성전자 주식회사
Applicant(s)

2001 년 02 월 27 일

특 허 청
COMMISSIONER

【서류명】 특허출원서

【권리구분】 특허

【수신처】 특허청장

【제출일자】 2001.02.13

【발명의 명칭】 버츄얼 공통패턴을 사용한 저전력 CSD 선형위상 FIR 필터 구조 및 그에 따른 필터구현방법

【발명의 영문명칭】 Low power CSD Linear phase FIR filter architecture using virtual common subexpression and filter design method therefore

【출원인】

【명칭】 삼성전자 주식회사

【출원인코드】 1-1998-104271-3

【대리인】

【성명】 김능균

【대리인코드】 9-1998-000109-0

【포괄위임등록번호】 1999-005679-8

【발명자】

【성명의 국문표기】 장영범

【성명의 영문표기】 JANG,Young Beom

【주민등록번호】 580808-1019111

【우편번호】 120-170

【주소】 서울특별시 서대문구 대현동 11-1

【국적】 KR

【신규성주장】

【공개형태】 간행물 발표(한국통신 학회 논문지)

【공개일자】 2000.12.31

【심사청구】 청구

【취지】 특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인
김능균 (인)

【수수료】

【기본출원료】 20 면 29,000 원

【가산출원료】 25 면 25,000 원

【우선권주장료】	0 건	0 원
【심사청구료】	13 항	525,000 원
【합계】	579,000 원	
【첨부서류】	1. 요약서·명세서(도면)_1통 2.신규성(출원시의 특례)규정을 적용받기 위한 증명서류_1통[추후제 출]	

【요약서】

【요약】

덧셈기의 사용을 최소화하여 고속 저전력 연산을 제공하는 개선된 디지털 필터가 개시된다. n 비트(n은 2이상의 자연수)의 CSD 코드워드로 표현되는 필터계수들을 복수로 갖는 디지털 필터에서 상기 필터계수들을 구현하는 방법은, 상기 필터계수들의 임의의 필터계수들에 대한 코드워드 패턴들 중에서, 정해진 공통패턴(common subexpression)의 비트이동, 비트가산, 비트반전을 통하여 같아지는 패턴들을 찾아서 버츄얼(virtual) 공통패턴으로 만들어 상기 임의의 필터계수들의 탭라인에서 상기 버츄얼 공통패턴이 공통패턴을 이용하여 수행됨으로서 덧셈의 사용 수를 감소시키는 방법을 특징으로 한다.

【대표도】

도 4

【색인어】

FIR 필터, CSD, 공통패턴, 버츄얼 공통패턴, 필터구현방법

【명세서】

【발명의 명칭】

버츄얼 공통패턴을 사용한 저전력 ＣＳＤ 선형위상 ＦＩＲ 필터 구조 및 그에 따른 필터구현방법{Low power CSD Linear phase FIR filter architecture using virtual common subexpression and filter design method therefore}

【도면의 간단한 설명】

도 1은 종래의 공통패턴을 사용한 CSD 선형위상 필터구조도

도 2는 종래의 공통패턴을 사용한 CSD 선형위상 필터구조도

도 3은 종래의 공통패턴을 사용한 CSD 선형위상 필터구조도

도 4는 본 발명의 일 예에 따라 도 1을 개선한 CSD 선형위상 필터구조도

도 5는 본 발명에 일 예에 따라 도 2를 개선한 CSD 선형위상 필터구조도

도 6은 본 발명에 일 예에 따라 도 3을 개선한 CSD 선형위상 필터구조도

도 7은 본 발명이 적용될 수 있는 무선 수신단말기의 IF 디지털 신호처리부의 블록도

도 8 및 도 9는 본 발명에 따른 버츄얼 공통패턴 및 종래의 공통패턴을 이용하여 나타낸 필터의 패턴구조도들

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<9> 본 발명은 디지털 필터에 관한 것으로, 특히 공통패턴을 사용한 저전력 CSD 필터 구조 및 그에 따른 필터 구현방법에 관한 것이다.

<10> 일반적으로 디지털 필터는 곱셈기, 덧셈기, 및 지연소자를 포함하는 회로들을 반도체 기판 상에 제조하는 것에 의해 구현될 수 있다. 그러한 디지털 필터의 입력신호와 필터계수 그리고 출력신호는 모두 2의 보수형(2's complement)으로 표현되는 것이 일반적이다. 고속 저전력 필터가 이동 통신 시스템 단말기 등의 IF (Intermediate Frequency) 다운 컨버터 칩에서 요구되어짐에 따라, 곱셈기를 사용하지 않고 덧셈기 만으로 곱셈을 수행하는 방법이 많이 사용되고 있다.

<11> 그러한 덧셈과 지연소자만으로 구현되는 디지털 필터에서는 2의 보수형 보다는 CSD(Canonical Signed Digit:이하 CSD)형의 표현이 널리 사용된다. 왜냐하면 임의의 필터계수를 바이너리(Binary)로 나타낼 때, 2의 보수형보다는 CSD형이 1의 수가 적게 사용되기 때문이다. 단 CSD형의 표현에서는 1과 -1이 모두 사용되는 점이 2의 보수형의 표현과는 다르다. CSD형의 표현에서 구현 측면을 보면, 1의 구현 시에는 덧셈기가 사용되고, -1의 구현에서는 뺄셈기가 사용되나 회로 구현비용은 같다.

<12> 일반적으로, 필터계수에 사용되는 1 또는 -1의 수가 덧셈의 수와 일치한다. 즉, 임의의 필터에서 1 또는 -1이 총 m개가 있을 때, 구현을 위하여 m-1개의 덧셈이 필요하다. 예를 들어, m=24 인 경우에 23개의 덧셈기가 필요하다.

<13> 그러나, 최근에는 CSD형의 필터구현에서 공통패턴을 공유함으로서 덧셈의 수를 감소시킨 방법이 사용되고 있다. 선형위상(Linear phase) FIR 필터에서, 예컨대 8개의 계수들이 있을 때 h0~h3의 계수는 h7~h4의 계수와 각각 대칭이다. 이와 같이 선형 위상 FIR 필터는 자연적으로 공통패턴이 2개 이상 존재한다.

<14> 이하에서는 후술될 본 발명의 철저한 이해를 제공할 의도외에는 다른 의도없이, 필터 계수(coefficients)의 대칭에서 생성되는 공통패턴을 이용하여 FIR(Finite Impulse Response:이하 FIR)필터를 구현하는 종래 기술의 예를 3가지 들고 각기 설명한다.

<15> 첫째의 경우로서, 다음과 같은 14탭의 선형위상 FIR 필터를 살펴보기로 하자.

<16> $H(z) = h_0 + h_1z^{-1} + h_2z^{-2} + h_3z^{-3} + h_4z^{-4} + h_5z^{-5} + h_6z^{-6} + h_7z^{-7} + h_8z^{-8} + h_9z^{-9} + h_{10}z^{-10} + h_{11}z^{-11} + h_{12}z^{-12} + h_{13}z^{-13}$

<17> 위의 필터계수가 다음과 같은 CSD형으로 나타내진다고 하자. 다음의 표-1에서 -1은 N으로 표기된다.

<18> 표-1

<19>

	-1	-2	-3	-4	-5	-6	-7	-8	-9
h0		1			N			1	
h1		1			N				1
h2		1			N		1		
h3		1				N		1	
h4		1		N				1	
h5			1		N			1	
h6	1				N			1	

<20> 위의 표-1에서 0은 별도로 표기되지 않고 블랭크로 나타내었으며, h7부터 h13까지의 계수들은 h0부터 h6까지의 계수들과 대칭이므로 생략되었다. 위의 표에서 가로방향은 9비트로 표현되는 CSD형의 필터계수를 나타내며, 세로방향은 필터 계수를 나타낸다. 즉

h0의 필터계수는 0100N0010의 CSD 형의 계수이다. 상기 표에서 가로방향은 쉬프트를 의미하고, 세로는 지연을 의미한다. 1번째 행과 1번째 열은 x_1으로 정의되며, 기준 점으로 정해진다.

<21> 위의 각각의 계수 패턴들을 수식으로 나타내면 다음과 같다.

<22> $h_0 : x_2 = x_1 - x_1 \gg 3 + x_1 \gg 6$

<23> $h_1 : x_3 = x_1 - x_1 \gg 3 + x_1 \gg 7$

<24> $h_2 : x_4 = x_1 - x_1 \gg 3 + x_1 \gg 5$

<25> $h_3 : x_5 = x_1 - x_1 \gg 4 + x_1 \gg 6$

<26> $h_4 : x_6 = x_1 - x_1 \gg 2 + x_1 \gg 6$

<27> $h_5 : x_7 = x_1 \gg 1 - x_1 \gg 3 + x_1 \gg 6$

<28> $h_6 : x_8 = x_1 \gg (-1) - x_1 \gg 3 + x_1 \gg 6$

<29> 상기 수식에서 '>>'은 쉬프트를 나타내는 부호로 사용되었다. 예를 들어, 상기 $h_0 : x_2 = x_1 - x_1 \gg 3 + x_1 \gg 6$ 라는 표현은 x_1 을 기준으로 정할 때, 표-1에서 보여지는 바와 같이 $-x_1 \gg 3$은 상기 x_1을 기준으로 3 비트 쉬프트, $x_1 \gg 6$ 은 상기 x_1 을 기준으로 6비트 쉬프트를 의미한다. 여기서, 상기 x_1 은 1비트 쉬프트를 자체로 내재하고 있다. 따라서, 쉬프트되는 전체 비트수는 각기 -1,-(-4),-7로서 나타난다. 상기와 같은 방식으로 상기 $h_1 : x_3 = x_1 - x_1 \gg 3 + x_1 \gg 7$ 은 -1,-4,-8이 된다. 또한, 상기 -(-4)는 +4인데 이는 표-1에서 N으로 되어 있기 때문이며, 실제로는 뺄셈기의 사용을 필요로 한다.

<30> 위의 패턴을 사용하여 출력신호를 표기하면 다음과 같다.

<31> $y = x_2 >> 1 + x_3[-1] >> 1 + x_4[-2] >> 1 + x_5[-3] >> 1 + x_6[-4] >> 1 + x_7[-5] >> 1 + symmetry$

<32> 결국 출력은 입력신호 x의 쉬프트(shift)되고 지연된 여러 개의 합으로 표현될 수 있는 것이다. 상기 symmetry 는 표기의 간략화를 위해 사용된 용어로서 대칭의 계수가 반복되는 것을 의미한다. 즉, h7부터 h13 까지의 계수는 h0부터 h6까지의 계수와 대칭이므로 별도의 표시를 생략하고 symmetry로서 표기되었다. 상기한 바와 같은 계수 패턴들을 가지는 FIR 필터를 트랜스포즈드 다이랙트 폼(transposed direct form)이용하여 구현하면 도 1과 같은 구성이 얻어진다.

<33> 도 1을 참조하면, 라인(T1~T7)은 탭들(taps)을 나타내며, 덧셈기는 부호 Ai(i는 1 이상의 자연수)로 표기되고, 지연기는 부호 Di로 표기되며, 입력신호는 x, 출력신호는 y로 나타나 있다. 도면에서 비록 덧셈기로 표기되었으나 뺄샘을 행하는 뺄셈기는 그 입력단에 부호'-'가 표기되어 있다. 본 명세서에서 사용되는 용어 '덧셈기'는 덧셈을 행하는 기능을 가짐은 물론, 실질적으로 뺄셈을 수행하는 뺄셈기와 동일한 기능을 갖는 의미로 사용됨을 유의하여야 한다.

<34> 도 1을 참조하면, 탭라인(T1)에서 보여지는 덧셈기들(A1,A2)에 의해 -1,-4,-7의 덧셈이 행하여져 상기 h_0 계수가 구현되고, 탭라인(T2)에서 보여지는 덧셈기들(A3,A4)에 의해 -1,-4,-8의 덧셈이 행하여져 상기 h_1 계수가 구현된다. 상기 -1,-4,-8과 같이 -n(n은 정수)은 2^{-n}을 의미하며 n비트 라이트(right)시프트 동작이다. 결국, 입력신호 x는 덧셈이 수행되기 이전에 -1,-4,-8과 곱하여지며 이는 시프트 레지스터에 의해 구현된다. 즉 도면에서의 -n은 시프트 레지스터의 하드웨어로 구현됨을 의미한다.

<35> 지연기들(D1~D13)은 출력신호 y를 얻기 위해 각기 대응되는 덧셈기의 출력을 주어

진 타임 만큼 지연하기 위한 기능을 한다. 상기 h_0 계수 구현시 -1,-4,-7로 되는 것은 x_1자체가 -1의 쉬프트를 내재하고 있는 기준 점이므로 x_1을 구현할 때에는 구현 식보다 -1을 더 쉬프트 하여야 하기 때문이다.

<36> 도 1과 같은 필터를 구현하는데 사용된 덧셈의 수를 보면 다음과 같다. 먼저 각각의 필터계수를 만드는데 14개의 덧셈기들(A1~A14)이 필요하다. 왜냐하면 총 7개의 계수가 필요한데 한 개의 필터 계수당 2개의 덧셈이 사용되기 때문이다. 그리고 출력신호를 뽑아 내기 위하여 13개의 덧셈기들(A12~A27)이 필요하다. 따라서 총 14+13=27개의 덧셈기가 사용됨을 알 수 있다.

<37> 상기 표-1에서 보듯이 h0의 패턴과 나머지 계수들의 패턴이 유사하나 공통패턴은 될 수 없으므로 덧셈의 감소는 성취될 수 없다. 이와 같은 패턴에서 덧셈의 감소를 달성할 수 있는 방법이 모색되어야만 CSD형의 필터에서 저전력 및 고속의 필터가 구현될 수 있다.

<38> 상기한 바와 같이, 도 1과 같은 필터구성에서 필터 계수(coefficients)의 대칭에서 생성되는 공통패턴을 이용하여 덧셈기의 수를 어느 정도 감소시킬 수 는 있으나, 여전히 덧셈기가 많이 소요되는 문제점이 있다.

<39> 두번째의 경우로서, 다음과 같은 10탭의 선형위상 FIR 필터를 살펴보기로 하자.

<40> $H(z) = h_0 + h_1z^{-1} + h_2z^{-2} + h_3z^{-3} + h_4z^{-4} + h_5z^{-5} + h_6z^{-6} + h_7z^{-7} + h_8z^{-8} + h_9z^{-9}$

<41> 위의 필터의 계수가 다음과 같은 CSD형으로 나타내진다고 하자. 다음의 표-2에서 -1은 N으로 표기된다.

<42> 표-2

<43>

	-1	-2	-3	-4	-5	-6	-7	-8	-9
h0	1		N			1		N	
h1			N			1		N	
h2	1					1		N	
h3	1		N					N	
h4	1		N			1			

<44> h5부터 h9까지의 계수들은 h0부터 h4까지의 계수들과 대칭이므로 생략하였다. 위의 표에서 가로는 9비트로 표현되는 CSD형의 필터계수를 나타내며, 세로는 필터 계수를 나타낸다. 즉 h0의 계수는 10N0010N0의 CSD 형의 계수이다. 위의 각각의 계수 패턴들을 수식으로 나타내면 다음과 같다.

<45> $h_0 : x_2 = x_1 - x_1 >> 2 + x_1 >> 5 - x_1 >> 7$

<46> $h_1 : x_3 = -x_1 >> 2 + x_1 >> 5 - x_1 >> 7$

<47> $h_2 : x_4 = x_1 + x_1 >> 5 - x_1 >> 7$

<48> $h_3 : x_5 = x_1 - x_1 >> 2 - x_1 >> 7$

<49> $h_4 : x_6 = x_1 - x_1 >> 2 + x_1 >> 5$

<50> 위의 패턴을 사용하여 출력신호를 표기하면 다음과 같다.

<51> $y = x_2 + x_3[-1] + x_4[-2] + x_5[-3] + x_6[-4] + \text{symmetry}$

<52> 상기한 바와 같은 계수 패턴들을 가지는 FIR 필터를 트랜스포즈드 다이렉트 폼을 이용하여 구현하면 도 2와 같은 구성이 얻어진다.

<53> 도 2를 참조하면, 도 1과 동일 또는 유사하게 라인(T1~T5)은 탭들(taps)을 나타내며, 덧셈기는 부호 Ai(i는 1이상의 자연수)로 표기되고, 지연기는 부호 Di로 표기되며, 입력신호는 x, 출력신호는 y로 나타나 있다. 도면에서 비록 덧셈기로 표기되었으나 뺄샘을 행하는 뺄셈기는 그 입력단에 부호'-'가 표기되어 있다. 탭라인(T1)에서 보여지는 덧

셈기들(A1,A2,A3)에 의해 -1,-3,-6,-8의 덧셈이 행하여져 상기 h_0 계수가 구현되고, 탭 라인(T2)에서 보여지는 덧셈기들(A4,A5)에 의해 -3,-6,-8의 덧셈이 행하여져 상기 h_1 계수가 구현된다. 지연기들(D1~D9)은 출력신호 y를 얻기 위해 각기 대응되는 덧셈기의 출력을 주어진 타임 만큼 지연하기 위한 기능을 한다.

<54> 도 2에서, 필터를 구현하는데 필요한 덧셈의 수를 보면 다음과 같다. 먼저 각각의 필터계수를 만드는데 11개의 덧셈기들(A1~A11)이 필요하다. 왜냐하면 각각의 계수에 3, 2, 2, 2, 2개의 덧셈이 사용되기 때문이다. 그리고 지연기들에 연결되어 덧셈을 행하는 덧셈기들(A12~A20)이 9개 필요하다. 따라서 총 11+9=20개의 덧셈기들이 필요하다.

<55> 표-2에서 보듯이 h0의 패턴과 나머지 계수들의 패턴이 유사하나 공통패턴은 될 수 없으므로 역시 덧셈의 감소는 성취될 수 없다. 이와 같은 패턴에서도 덧셈의 감소를 달성할 수 있는 방법이 모색되어야만 CSD형의 필터에서 저전력으로 필터 구현이 가능해 질 수 있다.

<56> 상기한 바와 같이, 도 2와 같은 필터구성에서 필터 계수의 대칭에서 생성되는 공통패턴을 이용하여 덧셈기의 수를 어느 정도 감소시킬 수 는 있으나, 여전히 덧셈기가 많이 소요되는 문제점에 기인하여 고속 및 저전력 특성을 최적으로 달성하기 어렵다.

<57> 이제, 세 번째의 경우로서, 다음과 같은 10탭의 선형위상 FIR 필터를 살펴보기로 하자.

<58> $H(z) = h_0 + h_1z^{-1} + h_2z^{-2} + h_3z^{-3} + h_4z^{-4} + h_5z^{-5} + h_6z^{-6} + h_7z^{-7} + h_8z^{-8} + h_9z^{-9}$

<59> 위의 필터의 계수가 다음과 같은 CSD형으로 나타내진다고 하자. 다음의 표-3에서 -1은 N으로 표기된다.

<60> 표-3

<61>

	-1	-2	-3	-4	-5	-6	-7	-8	-9
h0	1		N			1		N	
h1	N		N			1		N	
h2	1		1			1		N	
h3	1		N			N		N	
h4	1		N			1		1	

<62> 상기의 표-3에서 h5부터 h9까지의 계수들은 h0부터 h4까지의 계수들과 대칭이므로 생략하였다. 위의 표에서 가로는 9비트로 표현되는 CSD형의 필터계수를 나타내며, 세로는 필터 계수를 나타낸다. 즉 h0의 계수는 10N0010N0의 CSD 형의 계수이다.

<63> 위의 각각의 계수 패턴들을 수식으로 나타내면 다음과 같다.

<64> $h_0 : x_2 = x_1 - x_1 >> 2 + x_1 >> 5 - x_1 >> 7$

<65> $h_1 : x_3 = -x_1 - x_1 >> 2 + x_1 >> 5 - x_1 >> 7$

<66> $h_2 : x_4 = x_1 + x_1 >> 2 + x_1 >> 5 - x_1 >> 7$

<67> $h_3 : x_5 = x_1 - x_1 >> 2 - x_1 >> 5 - x_1 >> 7$

<68> $h_4 : x_6 = x_1 - x_1 >> 2 + x_1 >> 5 + x_1 >> 7$

<69> 위의 패턴을 사용하여 출력신호를 표기하면 다음과 같다.

<70> $y = x_2 + x_3[-1] + x_4[-2] + x_5[-3] + x_6[-4] + symmetry$

<71> 상기한 바와 같은 계수 패턴들을 가지는 FIR 필터를 트랜스포즈드 다이랙트 폼을 이용하여 구현하면 도 3과 같은 구성이 얻어진다.

<72> 도 3을 참조하면, 도 1 또는 도 2의 경우와 동일 또는 유사하게 라인(T1~T5)은 탭들(taps)을 나타내며, 덧셈기는 부호 Ai(i는 1이상의 자연수)로 표기되고, 지연기는 부

호 Di로 표기되며, 입력신호는 x, 출력신호는 y로 나타나 있다. 탭라인(T1)에서 보여지는 덧셈기들(A1,A2,A3)에 의해 -1,-3,-6,-8의 덧셈이 행하여져 상기 h_0 계수가 구현되고, 탭라인(T2)에서 보여지는 덧셈기들(A4,A5,A6)에 의해 -1,-3,-6,-8의 덧셈이 행하여져 상기 h_1 계수가 구현된다. 지연기들(D1~D9)은 출력신호 y를 얻기 위해 각기 대응되는 덧셈기의 출력을 주어진 타임 만큼 지연하기 위한 기능을 한다.

<73> 도 3에서, 필터를 구현하는데 필요한 덧셈의 수를 보면 다음과 같다. 먼저 각각의 필터계수를 만드는데 15개의 덧셈이 필요하다. 왜냐하면 각각의 계수를 구현하기 위해 3개씩의 덧셈기들이 5개의 탭들에 사용되기 때문이다. 그리고 지연기들에 연결되어 덧셈을 행하는 덧셈기가 9개 필요하다. 따라서 총 15+9=24개의 덧셈기가 사용된다.

<74> 표-3에서 보듯이 h0의 패턴과 나머지 계수들의 패턴이 유사하나 역시 공통패턴은 될 수 없으므로 덧셈의 감소는 성취될 수 없다. 이와 같은 패턴에서도 덧셈의 감소를 달성할 수 있는 방법이 모색되어야만 CSD형의 필터에서 저전력으로 필터 구현이 가능해 질 수 있다.

<75> 도 3와 같은 필터구성에서도 필터 계수의 대칭에서 생성되는 공통패턴을 이용하여 덧셈기의 수를 어느 정도 감소시킬 수 는 있으나, 여전히 덧셈기가 많이 소요되는 문제점에 기인하여 고속 및 저전력 특성을 최적으로 달성하기 어렵다.

<76> 상기한 바와 같이 선형위상 FIR 필터에서는 계수의 대칭에서 생성되는 공통패턴을 이용하여 덧셈기를 줄이는 기법이 우선적으로 사용되며, 일반 FIR 필터에서는 계수의 대칭에서 생성되는 공통패턴이 없으므로 계수들에서 나타나는 공통패턴을 조사하여 공유하는 기법이 사용된다. 두 가지 기법 모두가 필터계수 내에 존재하는 공통패턴을 공유하는 방법이라는 점에서는 같은 개념이다.

<77> 이와 같이 CSD(Canonical Signed Digit)를 사용한 FIR 필터구현에서, 사용되어지는 가산기의 수를 줄이기 위한 공통패턴 공유 방법이 서브 익스프레션 셰어링 (Subexpression sharing)기술이라는 명칭으로 본 분야에서 개시되어졌다. 이는 리챠드 아이 하틀리(Richard I. Hartley)에 의해 발표된 제목 'Subexpression sharing in filters using canonic signed digit multipliers'하에 IEEE Transaction on circuits and systems II: Analog and digital signal processing, Vol. 43, No. 10, pp. 677-688, October 1996. 에 나타나 있다.

<78> 또한, 최소의 가산기를 이용하는 고속 FIR 디지털 필터 구조와 믹스드 인테저 프로그래밍(MILP)을 사용한 필터의 설계방법에 관한 기술이 M. Yagyu, A. Nishihara, and N. Fujii, 'Fast FIR digital filter structures using minimal number of adders and its application to filter design', IEICE Transaction on Fundamentals, Vol. E79 A, No. 8, pp. 1120-1129, August 1996.에 개시된 바 있다.

<79> 상기한 선행기술들에서는 선형위상 필터 계수 대칭에서 생성되는 공통패턴이외에 계수 내부에 존재하는 공통패턴들을 찾아서 공유함으로써 덧셈기의 수를 줄이고 있으나, 고속 저전력의 반도체 칩을 구현할 만큼 덧셈기의 수를 감소시키지 못하는 문제점이 있다.

<80> 상술한 바와 같이 종래에는 덧셈기의 수가 많아서 고속처리 및 저전력 구현이 어려운 문제점이 있어 왔다. CSD형으로 구현되는 디지털 필터에서 덧셈기의 수가 반도체의 구현면적과 처리속도를 정의하므로, 덧셈기의 수를 보다 최소화할 수 있는 기술이 절실히 필요하게 된다.

<81> 따라서, 상기한 종래의 문제점들을 해결할 수 있는 개선된 기술이 본 분야에서 절

실히 요망되는 실정이다.

【발명이 이루고자 하는 기술적 과제】

<82> 따라서, 본 발명의 목적은 상기한 종래의 문제점들을 해소할 수 있는 디지털 필터를 제공함에 있다.

<83> 본 발명의 다른 목적은 공통패턴을 최대로 이용하여 덧셈 연산을 최소화하는 저전력 CSD 선형위상 필터 구조 및 그에 따른 필터구현방법을 제공함에 있다.]

<84> 본 발명의 또 다른 목적은 저전력 고속 동작의 FIR필터를 제공함에 있다.

<85> 본 발명의 또 다른 목적은 보다 간단한 회로 구성으로 칩내의 점유면적을 최소화하고 제조원가를 저렴하게 할 수 있는 저전력 고속 동작의 디지털 필터 및 그에 따른 필터 구현방법을 제공함에 있다.

<86> 상기한 목적들 및 타의 목적을 달성하기 위한 본 발명의 일 양상(aspect)에 따라, n 비트(n은 2이상의 자연수)의 CSD 코드워드로 표현되는 필터계수들을 복수로 갖는 디지털 필터에서 상기 필터계수들을 구현하는 방법은, 상기 필터계수들 중 필터계수들 안에 나타나는 임의의 코드워드 패턴을 기존의 공통패턴을 이용하여 구현함으로서 구현 비용을 절감시키는 것을 특징으로 한다.

<87> 본 발명의 다른 양상에 따라, 디지털 필터는: k 비트의 디지털 샘플들을 입력신호로서 수신하여 CSD 코드내에서 n 비트의 코드워드로서 각기 표현되는 필터 계수들중에서 공통패턴으로 정의된 필터 계수의 비트 쉬프트 값들 만큼 각기 쉬프팅하는 제1쉬프트 레지스터 멤버들과, 상기 공통패턴으로 정의되지 않은 다른 필터 계수들의 코드워드를

상기 공통패턴의 코드워드를 이용하여 구현하기 위하여 쉬프팅하는 제2쉬프트 레지스터 멤버들을 포함하는 쉬프트 레지스터 그룹과; 상기 제1쉬프트 레지스터 멤버들로부터 출력된 쉬프팅된 디지털 샘플들을 모두 더하여 공통 탭라인에 제공하는 제1 합성멤버들과, 상기 제2쉬프트 레지스터 멤버들로부터 출력된 쉬프팅된 디지털 샘플들을 상기 공통 탭라인의 합성출력과 각기 더하여 각각의 대응되는 탭라인에 제공하는 제2 합성멤버들을 포함하는 덧셈그룹과; 상기 탭라인들에 연결되며 상기 합성출력들 간에 딜레이를 제공하기 위해 직렬로 연결된 복수의 지연기를 가지는 지연그룹과; 상기 지연기들의 출력과 상기 탭라인의 합성출력들을 더하여 k비트의 디지털 출력신호를 생성하기 위한 복수의 덧셈기를 포함하는 출력 덧셈그룹을 적어도 포함함을 특징으로 한다.

<88> 상기한 본 발명의 방법 및 필터구조에 따르면, 고속 저전력 연산 및 덧셈기의 사용 개수를 보다 최소화 할 수 있다.

【발명의 구성 및 작용】

<89> 상기한 본 발명의 목적들 및 타의 목적들, 특징, 그리고 이점들은, 첨부된 도면들을 참조하여 이하에서 기술되는 본 발명의 상세하고 바람직한 실시예의 설명에 의해 보다 명확해질 것이다. 도면들 내에서 서로 동일 내지 유사한 부분들은 설명 및 이해의 편의상 동일 내지 유사한 참조부호들로 기재됨을 주목하여야 한다.

<90> 먼저, 본 발명의 실시예들의 전반에 걸쳐 일관되게 견지되고 있는 기술적 사상은, 처리할 임의의 패턴이 공통패턴에 대하여 필터계수의 대칭이나 보수형태로도 나타나지 아니하는 임의의 패턴도 기존의 공통패턴을 이용하여 구현할 수 있다는 것이다. 예컨대,

상기 공통패턴에 대하여 비트이동(쉬프트), 비트가산, 또는 비트반전을 행하면 임의의 패턴과 같아질 수 있는데, 이를 본 명세서에서는 버츄얼 공통패턴이라고 명명하였다. 따라서, 버츄얼 공통패턴이라는 용어는, 원래는 공통패턴이 아닌 패턴이 기존의 공통패턴을 이용하여 간단히 구현될 수 있는 패턴의 의미로 해석되어야 한다.

<91> 본 발명에서 사용되는 덧셈기는 입력초단에 사용되는 특별한 경우를 제외하고는 모두 복수의 입력단들을 가지는 풀 애더(full adder)이다. 상기 풀 애더는 2개의 하프 애더(half adder)와 하나의 오아 게이트(OR gate)로 구현될 수 있다. 상기 하프 애더는 한 개의 배타적 논리합 게이트와 한 개의 앤드 게이트를 연결하여 구성될 수 있음은 명백하다. 지연기는 플립플롭 또는 레지스터로 구성될 수 있다.

<92> 먼저, 첫 번째의 경우로서, 비트 이동에 의한 버츄얼 공통패턴의 생성을 설명한다. 설명전후를 불문하고 도 4를 미리 참조하고 도 1과 비교하여 덧셈기의 감소를 카운팅 하여도 무방하다.

<93> 다음과 같은 14탭의 선형위상 FIR 필터를 살펴보기로 하자.

<94> $H(z) = h_0 + h_1z^{-1} + h_2z^{-2} + h_3z^{-3} + h_4z^{-4} + h_5z^{-5} + h_6z^{-6} + h_7z^{-7} + h_8z^{-8} + h_9z^{-9} + h_{10}z^{-10} + h_{11}z^{-11} + h_{12}z^{-12} + h_{13}z^{-13}$

<95> 위의 필터의 계수가 상기한 표-1에서와 같은 CSD형으로 나타내진다고 하자.

<96> 따라서, 상기 표-1에서 보여지는 각각의 계수 패턴들은 h0을 사용하여 다음과 같이 표기할 수 있다.

<97> h1 : 100N0001 = 100N001 - 00000001

<98> h2 : 100N010 = 100N001 + 0000001

<99> h3 : 1000N01 = 100N001 + 0000100

<100> h4 : 10N0001 = 100N001 - 0001000

<101> h5 : 010N001 = 100N001 - 0100000

<102> h6 : 1000N001 = 0100N001+01000000

<103> 즉, 모든 패턴은 공통패턴 h0인 100N001의 패턴을 이용하여 나타낼 수 있다. 위의 각각의 계수 패턴들을 수식으로 나타내면 다음과 같다.

<104> $h_0: x_2 = x_1 - x_1 >> 3 + x_1 >> 6$

<105> $h_1: x_3 = x_2 - x_1 >> 7$

<106> $h_2: x_4 = x_2 + x_1 >> 6$

<107> $h_3: x_5 = x_2 + x_1 >> 4$

<108> $h_4: x_6 = x_2 - x_1 >> 3$

<109> $h_5: x_7 = x_2 - x_1 >> 1$

<110> $h_6: x_8 = x_2 + x_1$

<111> 위의 패턴을 사용하여 출력신호를 표기하면 다음과 같다.

<112> $y = x_2 >> 1 + x_3[-1] >> 1 + x_4[-2] >> 1 + x_5[-3] >> 1 + x_6[-4] >> 1 + x_7[-5] >> 1 + x_8[-6] >> 1 + symmetry$

<113> 상기한 바와 같은 계수 패턴들을 가지는 FIR 필터를 트랜스포즈드 다이랙트 폼(transposed direct form)을 이용하여 구현하면 도 4와 같은 구성이 얻어진다.

<114> 도 4를 참조하면, 라인(T1~T7)은 탭들(taps)을 나타내며, 덧셈기는 부호 Ai(i는 1

이상의 자연수)로 표기되고, 지연기는 부호 Di로 표기되며, 입력신호는 x, 출력신호는 y로 나타나 있다. 도면에서 비록 덧셈기로 표기되었으나 뺄샘을 행하는 뺄셈기는 그 입력단에 부호 '-'가 표기되어 있다.

<115> 도 4를 참조하면, 탭라인(T1)에서 보여지는 덧셈기들(A1,A2)에 의해 -1,-4,-7의 덧셈이 행하여져 상기 h_0 계수가 구현되고, 탭라인(T2)에서 보여지는 덧셈기(A3)에 의해 h_0 계수와 -8의 덧셈이 행하여져 상기 h_1 계수가 구현된다. 탭라인(T3)에서 보여지는 덧셈기(A4)에 의해 h_0 계수와 -7의 덧셈이 행하여져 상기 h2 계수가 구현되고, 탭라인(T4)에서 보여지는 덧셈기(A5)에 의해 h_0 계수와 -5의 덧셈이 행하여져 상기 h3 계수가 구현되며, 탭라인(T5)에서 보여지는 덧셈기(A6)에 의해 h_0 계수와 -4의 덧셈이 행하여져 상기 h4 계수가 구현되고, 탭라인(T6)에서 보여지는 덧셈기(A7)에 의해 h_0 계수와 -2의 덧셈이 행하여져 상기 h5 계수가 구현되며, 탭라인(T7)에서 보여지는 덧셈기(A8)에 의해 h_0 계수와 -1의 덧셈이 행하여져 상기 h6 계수가 구현된다. 여기서, 상기 덧셈기들(A3,A4,A5,A6,A7,A8)의 일측 입력단은 공통패턴으로 작용하는 필터계수 h0의 탭라인(T1)에 공통으로 연결됨을 유의하라. 지연기들(D1~D13)은 출력신호 y를 얻기 위해 각기 대응되는 덧셈기의 출력을 주어진 타임 만큼 지연하기 위한 기능을 한다. 예컨대, 지연기(D1)는 상기 덧셈기(A1)의 출력을 저장하고 있다가 상기 덧셈기(A3)의 출력이 나오는 시점에서 덧셈기(A9)의 입력으로 제공한다. 지연기(D2)는 상기 덧셈기(A4)의 출력이 나올 때 저장된 상기 덧셈기(A9)의 출력을 덧셈기(A10)의 입력으로 제공한다.

<116> 도 4와 같은 필터를 구현하는데 사용된 덧셈의 수를 보면 다음과 같다. 먼저 각각의 필터계수를 만드는데 8개의 덧셈기들(A1~A8)이 사용되었다. 그리고 출력신호 y를 도

출하기 위하여 지연기들에 의해 지연된 출력과 탭라인에 나타나는 출력을 합하는 덧셈기들(A9~A21)이 13개 사용되었다. 따라서 총 8+13=21개의 덧셈기가 사용되었다. 이와 같이, 비트이동에 의하여 버츄얼 공통패턴을 생성하여 도 4와 같이 필터를 구성하면, 도 1과 같은 종래의 경우에 비해 6개의 덧셈기를 추가로 더 줄일 수 있다. 즉, 도 1에서는 27개의 덧셈기가 사용되었으나 도 4에서는 21개의 덧셈기가 사용되었음을 주목하라.

<117> 덧셈기가 줄어든 이유를 강조하여 설명하면, 종래의 방법에서는 h1부터 h6의 필터계수가 h0의 공통패턴과는 관계가 없으므로 h1부터 h6의 필터계수를 h0의 공통패턴을 이용하여 구현하지 못하고 그 자체로서만 구현하였다. 그러나, 본 발명에서는 종래의 방법과는 달리, 구현하고자 하는 임의의 패턴들 즉, h1부터 h6의 필터계수들을 모두 h0의 필터계수를 이용하여 표현하였다. 즉, 임의의 패턴이 공통패턴의 1비트 비트이동을 통하여 같아지는 버츄얼 공통패턴을 만드는 것이다. 위의 수식에서 보여지듯이 h1부터 h6의 필터계수들은 모두 x_2의 공통패턴에 1개의 덧셈만을 추가하여 표현된 것임을 알 수 있다.

<118> 결국, CSD 코드워드로 표현되는 필터계수들을 복수로 갖는 디지털 필터에서 상기 필터계수들을 구현하는 방법은, 상기 필터계수들중 임의의 필터계수들에 대한 코드워드 패턴을 기존에 정해진 공통패턴의 비트이동을 통하여 구현하는 것이다. 따라서, 상기 임의의 필터계수들의 탭라인에서 상기 공통패턴을 공유하게 되는 덧셈이 각기 수행되도록 하는 것에 의해 달성된다.

<119> 상기한 바와 같이, 본 발명의 첫 번째 실시 예의 경우에는 비트이동을 통한 버츄얼 공통패턴을 생성하여 필터 구현에 사용되는 가산기의 수를 도 1의 경우에 비해 6개를 감소시켰다. 따라서, 감소된 만큼의 저전력 및 고속 처리가 달성되는 효과가 있다.

<120> 두 번째의 경우로서, 비트가산에 의한 버츄얼 공통패턴의 생성을 설명한다. 유사하게 도 5를 미리 참조하고 도 2와 비교하여 덧셈기의 감소를 카운팅하여도 무방하다.

<121> 다음과 같은 10탭의 선형위상 FIR 필터를 살펴보기로 하자.

<122> $H(z) = h_0 + h_1z^{-1} + h_2z^{-2} + h_3z^{-3} + h_4z^{-4} + h_5z^{-5} + h_6z^{-6} + h_7z^{-7} + h_8z^{-8} + h_9z^{-9}$

<123> 위의 필터의 계수가 상기한 표-2에서와 같은 CSD형으로 나타내진다고 하자.

<124> 상기 표-2에서 보여지는 각각의 패턴들은 h0을 사용하여 다음과 같이 표기될 수 있다.

<125> h1 : 00N0010N = 10N0010N - 10000000

<126> h2 : 1000010N = 10N0010N + 00100000

<127> h3 : 10N0000N = 10N0010N - 00000100

<128> h4 : 10N00100 = 10N0010N + 00000001

<129> 즉 모든 패턴은 h0의 패턴을 이용하여 다음과 같이 수식으로 나타낼 수 있다.

<130> $h_0: x_2 = x_1 - x_1 >> 2 + x_1 >> 5 - x_1 >> 7$

<131> $h_1: x_3 = x_2 - x_1$

<132> $h_2: x_4 = x_2 + x_1 >> 2$

<133> $h_3: x_5 = x_2 - x_1 >> 5$

<134> $h_4: x_6 = x_2 + x_1 >> 7$

<135> 위의 패턴을 사용하여 출력신호를 표기하면 다음과 같다.

<136> $y = x_2 + x_3[-1] + x_4[-2] + x_5[-3] + x_6[-4] + symmetry$

<137> 상기한 바와 같은 계수 패턴들을 가지는 FIR 필터를 트랜스포즈드 다이랙트 폼을 이용하여 구현하면 도 5와 같은 구성이 얻어진다.

<138> 도 5를 참조하면, 라인(T1~T5)은 탭들(taps)을 나타내며, 덧셈기는 부호 Ai(i는 1 이상의 자연수)로 표기되고, 지연기는 부호 Di로 표기되며, 입력신호는 x, 출력신호는 y로 나타나 있다. 도면에서 비록 덧셈기로 표기되었으나 뺄샘을 행하는 뺄셈기는 그 입력단에 부호 '-'가 표기되어 있다.

<139> 도 5를 참조하면, 탭라인(T1)에서 보여지는 덧셈기들(A1,A2,A3)에 의해 -1,-3,-6,-8의 덧셈이 행하여져 상기 h_0 계수가 구현되고, 탭라인(T2)에서 보여지는 덧셈기(A4)에 의해 h_0 계수와 -1의 덧셈이 행하여져 상기 h_1 계수가 구현된다. 탭라인(T3)에서 보여지는 덧셈기(A5)에 의해 h_0 계수와 -3의 덧셈이 행하여져 상기 h2 계수가 구현되고, 탭라인(T4)에서 보여지는 덧셈기(A6)에 의해 h_0 계수와 -6의 덧셈이 행하여져 상기 h3 계수가 구현되며, 탭라인(T5)에서 보여지는 덧셈기(A7)에 의해 h_0 계수와 -8의 덧셈이 행하여져 상기 h4 계수가 구현된다.

<140> 지연기들(D1~D9)은 출력신호 y를 얻기 위해 각기 대응되는 덧셈기의 출력을 주어진 타임 만큼 지연하기 위한 기능을 한다. 예컨대, 지연기(D1)는 상기 덧셈기(A1)의 출력을 저장하고 있다가 상기 덧셈기(A4)의 출력이 나오는 시점에서 덧셈기(A8)의 입력으로 제공한다. 지연기(D2)는 상기 덧셈기(A5)의 출력이 나올 때 저장된 상기 덧셈기(A8)의 출력을 덧셈기(A9)의 입력으로서 제공한다. 여기서, 상기 덧셈기들(A4,A5,A6,A7)의 일측 입력단은 공통패턴으로 작용하는 필터계수 h0의 탭라인(T1)에 공통으로 연결됨을 주목(note)하라.

<141> 도 5와 같은 필터를 구현하는데 사용된 덧셈기의 수를 보면 다음과 같다. 먼저 각각의 필터계수를 만드는 데에는 각각의 계수별로 3, 1, 1, 1, 1개의 덧셈기가 사용되기 때문에 총 7개의 덧셈기들(A1~A7)이 사용되었다. 그리고, 출력신호를 뽑아 내는데 9개의 덧셈기들(A8~A16)이 사용됨을 알 수 있다. 따라서, 총 7+9=16개의 덧셈기들이 사용되었다.

<142> 이와 같이, 비트가산에 의하여 버츄얼 공통패턴을 생성하여 도 5와 같이 필터를 구성하면, 도 2와 같은 종래의 경우에 비해 4개의 덧셈기를 추가로 더 줄일 수 있다. 즉, 도 2에서는 20개의 덧셈기가 사용되었으나 도 5에서는 16개의 덧셈기가 사용되었음을 주목하라.

<143> 이와 같이, 덧셈기가 감소된 이유를 또 다시 강조하여 설명하면, 종래의 방법에서는 h1부터 h4까지의 필터계수가 h0의 공통패턴과는 관계가 없으므로 h1부터 h4까지의 필터계수를 h0의 공통패턴을 이용하여 구현하지 못하고 그 자체로서만 구현하였다.
그러나, 본 발명에서는 종래의 방법과는 달리, 구현하고자 하는 임의의 패턴들 즉, h1부터 h4까지의 필터계수들을 모두 h0의 필터계수를 이용하여 표현하였다. 즉, 기존의 공통패턴의 1비트 비트가산을 통하여 같아지는 임의의 패턴들을 버츄얼 공통패턴으로 정의하여 구현하는 것이다. 위의 수식에서 보여지듯이 h1부터 h4의 필터계수들은 모두 x_2의 공통패턴에 1개의 덧셈만을 추가하여 표현된 것임을 알 수 있다.

<144> 상기한 바와 같이, 본 발명의 두 번째 실시 예의 경우에는 비트가산을 통한 버츄얼 공통패턴을 생성하여 필터 구현에 사용되는 가산기의 수를 도 2의 경우에 비해 4개를 감소시켰다. 따라서, 감소된 만큼의 저전력 및 고속 처리가 달성되는 효과가 있다.

<145> 본 발명의 실시 예들 중의 세 번째의 경우로서, 비트 반전에 의한 버츄얼 공통패

턴의 생성을 이하에서 설명한다. 유사하게 도 6를 미리 참조하고 도 3과 비교하여도 상관없다.

<146> 다음과 같은 10탭의 선형위상 FIR 필터를 살펴보기로 하자.

<147> $H(z) = h_0 + h_1z^{-1} + h_2z^{-2} + h_3z^{-3} + h_4z^{-4} + h_5z^{-5} + h_6z^{-6} + h_7z^{-7} + h_8z^{-8} + h_9z^{-9}$

<148> 위의 필터의 계수가 상기한 표-3과 같은 CSD형으로 나타내진다고 하자.

<149> 상기 표-3에서 가로는 9비트로 표현되는 한 개의 CSD형의 필터계수를 나타내며, 세로는 사용되는 여러 개의 필터 계수를 나타낸다. 즉 h0의 계수는 10N0010N0의 CSD 형의 계수이다.

<150> 위의 각각의 패턴들은 h0을 사용하여 다음과 같이 표기될 수 있다.

<151> h1 : N0N0010N = 10N0010N -100000000

<152> h2 : 1010010N = 10N0010N + 01000000

<153> h3 : 10N00N0N = 10N0010N - 00001000

<154> h4 : 10N00101 = 10N0010N + 00000010

<155> 즉, 모든 패턴은 공통패턴이 되는 h0인 패턴을 이용하여 수식으로 다음과 같이 나타낼 수 있다.

<156> $h_0 : x_2 = x_1 - x_1 >> 2 + x_1 >> 5 - x_1 >> 7$

<157> $h_1 : x_3 = x_2 - x_1 >> (-1)$

<158> $h_2 : x_4 = x_2 + x_1 >> 1$

<159> $h_3 : x_5 = x_2 - x_1 >> 4$

<160> $h_4 : x_6 = x_2 + x_1 >> 6$

<161> 위의 패턴을 사용하여 출력신호를 표기하면 다음과 같다.

<162> $y = x_2 + x_3[-1] + x_4[-2] + x_5[-3] + x_6[-4] + symmetry$

<163> 상기한 바와 같은 계수 패턴들을 가지는 FIR 필터를 트랜스포즈드 다이랙트 폼을 이용하여 구현하면 도 6과 같은 구성이 얻어진다.

<164> 도 6을 참조하면, 라인(T1~T5)은 탭들(taps)을 나타내며, 덧셈기는 부호 Ai(i는 1 이상의 자연수)로 표기되고, 지연기는 부호 Di로 표기되며, 입력신호는 x, 출력신호는 y로 나타나 있다. 도면에서 비록 덧셈기로 표기되었으나 뺄샘을 행하는 뺄셈기는 그 입력단에 부호 '-'가 표기되어 있다.

<165> 도 6을 참조하면, 탭라인(T1)에서 보여지는 덧셈기들(A1,A2,A3)에 의해 -1,-3,-6,-8의 덧셈이 행하여져 상기 h_0 계수가 구현되고, 탭라인(T2)에서 보여지는 덧셈기(A4)에 의해 h_0 계수와 입력신호의 덧셈이 행하여져 상기 h_1 계수가 구현된다. 탭라인(T3)에서 보여지는 덧셈기(A5)에 의해 h_0 계수와 -2의 덧셈이 행하여져 상기 h2 계수가 구현되고, 탭라인(T4)에서 보여지는 덧셈기(A6)에 의해 h_0 계수와 -5의 덧셈이 행하여져 상기 h3 계수가 구현되며, 탭라인(T5)에서 보여지는 덧셈기(A7)에 의해 h_0 계수와 -7의 덧셈이 행하여져 상기 h4 계수가 구현된다.

<166> 지연기들(D1~D9)은 출력신호 y를 얻기 위해 각기 대응되는 덧셈기의 출력을 주어진 타임 만큼 지연하기 위한 기능을 한다. 예컨대, 지연기(D1)는 상기 덧셈기(A1)의 출력을 저장하고 있다가 상기 덧셈기(A4)의 출력이 나오는 시점에서 덧셈기(A8)의 입력으로 제공한다. 지연기(D2)는 상기 덧셈기(A5)의 출력이 나올 때 저장된 상기 덧셈기(A8)의 출력을 덧셈기(A9)의 입력으로서 제공한다. 여기서, 상기 덧셈기들(A4,A5,A6,A7)의 일측

입력단은 공통패턴으로 작용하는 필터계수 h0의 탭라인(T1)에 공통으로 연결됨을 주목(note)하라.

<167> 도 6과 같은 필터를 구현하는데 사용된 덧셈기의 수를 보면 다음과 같다. 먼저 각각의 필터계수를 만드는 데에는 각각의 계수별로 3, 1, 1, 1, 1개의 덧셈기가 사용되기 때문에 총 7개의 덧셈기들(A1~A7)이 사용되었다. 그리고, 출력신호를 뽑아 내는데 9개의 덧셈기들(A8~A16)이 사용됨을 알 수 있다. 따라서, 총 7+9=16개의 덧셈기들이 사용되었다.

<168> 이와 같이, 비트반전에 의하여 버츄얼 공통패턴을 생성하여 도 6와 같이 필터를 구성하면, 도 3과 같은 종래의 경우에 비해 8개의 덧셈기를 추가로 더 줄일 수 있다. 즉, 도 3에서는 24개의 덧셈기가 사용되었으나 도 6에서는 16개의 덧셈기가 사용되었음을 주목하라.

<169> 이와 같이, 덧셈기가 감소된 이유를 설명하면, 종래의 방법에서는 h1부터 h4까지의 필터계수가 h0의 공통패턴과는 관계가 없으므로 h1부터 h4까지의 필터계수를 h0의 공통패턴을 이용하지 못하고 자체로서만 구현하였다. 그러나, 본 발명에서는 종래의 방법과는 달리, 구현하고자 하는 임의의 패턴들 즉, h1부터 h4까지의 필터계수들을 모두 h0의 필터계수를 이용하여 표현하였다. 즉, 공통패턴의 1비트의 비트반전을 통하여 같아지는 패턴들을 찾아내어 이를 버츄얼 공통패턴으로 정의하고 공통패턴을 이용하여 구현하는 것이다. 위의 수식에서 보여지듯이 h1부터 h4의 필터계수들은 모두 x_2의 공통패턴에 1개의 덧셈만을 추가하여 표현된 것임을 알 수 있다.

<170> 상기한 바와 같이, 본 발명의 세 번째 실시 예의 경우에는 비트반전을 통한 버츄얼 공통패턴을 생성하여 필터 구현에 사용되는 가산기의 수를 도 3의 경우에 비해 8개를

감소시켰다. 따라서, 감소된 만큼의 저전력 및 고속 처리가 달성되는 효과가 있다.

<171> 상기한 바와 같이, 임의의 패턴에 대하여 비트이동, 비트가산, 또는 비트반전을 행하여 버츄얼 공통패턴을 만들어 필터를 구현하면 가산기의 수를 종래의 방법에 비해 훨씬 더 줄일 수 있다.

<172> 이하에서는 상기한 바와 같은 본 발명을 도 7에 적용할 경우에 버츄얼 공통패턴의 생성에 의한 가산기의 실질적인 감소를 설명한다. 도 7을 참조하면, RF 신호처리부(10)를 통해 수신된 아나로그 신호는 A/D 변환기(22)에 의해 소정 비트의 디지털 샘플들로 변환되어 제1,2 승산기(24,25)에 제공된다. 제1,2 필터부(26,27)는 각기 필터링된 I,Q 신호를 기저대역 신호처리부(30)로 출력하기 위해 주어진 사양으로 필터링을 행한다.

<173> 상기 도 7에서, CDMA 이동통신 단말기의 IF단 사양의 디지털 필터 즉, 상기 제1,2 필터부(26,27)를 CSD형 아키텍처로 제작시, 필터의 샘플링 주파수는 19.6608MHz, 패스밴드(Passband) 주파수는 630KHz, 패스밴드 리플(ripple)은 0.1dB, 스톱밴드(Stopband)주파수는 1.2288MHz, 스톱밴드 감쇄(Stopband attenuation)는 -40dB로 정하였다.

<174> 위와 같은 사양을 만족하는 72탭의 선형위상 FIR 필터를 이하에서 설명한다. 72개의 필터계수는 대칭이므로 36개만을 24 비트 프리시젼(bit precision)의 CSD형의 계수로 나타낼 경우에 종래의 공통패턴 방식에 따르면 도 9와 같이 나타난다. 그러나, 본 발명에 의한 버츄얼 공통패턴에 따르면 도 8과 같이 나타난다. 비교를 위한 도 9와 본 발명의 적용 예를 보인 도 8에서 -1은 n으로 표기된다.

<175> 도 8 및 도 9에서, 1 또는 -1의 수가 총 m=458개이므로 아무런 공통패턴을 사용하

지 않을 시 m-1=457개의 덧셈이 필요하다. 또한, 선형위상 대칭에 의한 공통패턴만을 이용하면 229-1+36 =264개의 덧셈이 필요하다. 그러나, 종래의 방법에 따라 공통패턴의 공유를 통하여 구현하면 도 9와 같이 18+179-1= 196개의 덧셈기가 필요하게 된다. 결국, 도 9에서는 공통패턴을 열은 검은 색의 블록내에 표시하였으며, 맨 우측의 열에 그 필터계수를 구현하기 위한 곱셈기의 수를 나타내었다. 따라서 196개의 덧셈기가 필요한 공통패턴을 도 9에서는 보여주고 있다.

<176> 그러나, 도 8에서는 다음과 같이 버츄얼 공통패턴이 구현된다.

<177> h1 : *00101 = *0101의 bit shift(-1)

<178> h3 : *0100001 = *0100*01의 bit add(-1)

<179> h7 : 10*00*0* = 10*0010*의 bit 반전(-2)

<180> h8 : 100100*0*= 100100*01의 bit 반전(-2)

<181> h12 : *000*0* = *000*01의 bit 반전(-1)

<182> h13 : 101001 = 10101의 bit shift(-1)

<183> h15 : *00*01 = *0*01의 bit shift(-1)

<184> h18 : *0*001 = *0*01의 bit shift(-1)

<185> h19 : 10*000* = 10*010*의 bit add(-1)

<186> h20 : *010001 = *010*01의 bit add(-1)

<187> h21 : *0010* = *010*의 bit shift(-1)

<188> h23 : *000101= *000*01의 bit 반전(-1)

<189> h26 : *0*00* = *0*0*의 bit shift(-1)

<190> h29 : 100100*0*= 100100*01의 bit 반전(-2)

<191> h30 : 10*0010* = 새로 공통패턴으로 정의(-3)

<192> h34 : 10*00* = 10*0*의 bit shift(-1)

<193> Total: -21개의 세이브(save)

<194> 도 8에서와 같이, 본 발명의 적용 예에서는 총 36개의 계수 중에서 위와 같이 16개의 계수에서 총 21개의 덧셈 감소를 이룰 수 있으며, 이는 도 8에서 괄호부호로 표시되어 있다. 또한, 도 8의 맨 우측 열에는 도 9에 비하여 감소된 덧셈의 수가 기록되어 있으며, 이 열의 합은 158이다. 결국, 본 발명의 적용 예에서는 공통패턴을 만드는데 21개의 덧셈이 필요하므로 총 21+158-1=178의 덧셈기만이 필요하다.

<195> 따라서, 도 8을 도 9와 대비하여 참조시 종래의 공통패턴 방식에 비하여 총 18개의 덧셈이 감소됨을 알 수 있다. 지금까지의 결과를 표-4에 요약하였다.

<196> 표-4

<197>

구 분	덧셈의 수	%
직접구현 (m=458)	457	233.2
계수대칭 이용 방식	264	134.7
종래의 공통패턴 방식(도	196	100
본 발명의 버츄얼 공통패턴 방식 (도 8)	178	90.8

<198> 상기한 바와 같이, CDMA 이동통신 단말기의 IF단의 필터를 본 발명에 따른 방식의 아키텍처로 구현시, 기존의 공통패턴을 사용한 CSD 아키텍처보다 9.2%의 덧셈 감소의 효과를 얻을 수 있다.

<199> 본 발명에 따른 버츄얼 공통패턴 아키텍처는 선형 위상 FIR 필터뿐만 아니라 일반적인 필터에서도 사용될 수 있다.

<200> 상기한 바와 같이, 본 발명은 도면을 기준으로 예를 들어 기술되었지만 이에 한정되지 않으며 발명의 기술적 사상을 벗어나지 않는 범위 내에서 본 발명이 속하는 기술분야에서 통상의 지식을 갖는 자에 의해 다양한 변화와 변경이 가능함은 물론이다. 예를 들어, 필터 탭라인의 수 및 공통패턴의 설정과 비트 쉬프트, 비트 가산 및 반전에 의한 버츄얼 공통패턴의 설정을 사안에 따라 변경하거나, 가감할 수 있음은 물론이다. 또한, 곱셈을 위한 쉬프트 레지스터, 덧셈기, 지연기들을 하드웨어적으로 구현하는 것 이외에 마이크로프로세서나 디지털 신호처리기에 의해 소프트웨어적으로 필터를 구현할 수 있다.

【발명의 효과】

<201> 상술한 바와 같이, 본 발명에 따르면 필터구현에 사용되는 덧셈의 수를 최소화하여 고속 및 저전력 디지털 필터를 제공하는 효과가 있다.

【특허청구범위】

【청구항 1】

n 비트(n은 2이상의 자연수)의 CSD 코드워드로 표현되는 필터계수들을 복수로 갖는 디지털 필터에서 상기 필터계수들을 구현하는 방법에 있어서,

상기 필터계수들중 임의의 필터계수들에 대한 코드워드 패턴을 정해진 공통패턴과 관련성 있는 버츄얼 공통패턴으로 만드는 것에 의해 상기 임의의 필터계수들의 탭라인에서 상기 공통패턴을 공유하게 되는 덧셈이 각기 수행되도록 하는 것을 특징으로 하는 방법.

【청구항 2】

제1항에 있어서, 상기 버츄얼 공통패턴은 비트이동에 의해 상기 공통패턴과 같아지게 됨을 특징으로 하는 방법.

【청구항 3】

제1항에 있어서, 상기 버츄얼 공통패턴은 비트가산에 의해 상기 공통패턴과 같아지게 됨을 특징으로 하는 방법.

【청구항 4】

제1항에 있어서, 상기 버츄얼 공통패턴은 비트반전에 의해 상기 공통패턴과 같아지게 됨을 특징으로 하는 방법.

【청구항 5】

제1항에 있어서, 상기 버츄얼 공통패턴은 비트이동, 비트가산, 또는 비트반전중의 적어도 둘 이상을 행함에 의해 상기 공통패턴과 같아짐을 특징으로 하는 방법.

【청구항 6】

제1항에 있어서, 상기 디지털 필터는 선형위상 FIR 필터임을 특징으로 하는 방법.

【청구항 7】

n 비트(n은 2이상의 자연수)의 CSD 코드워드로 표현되는 필터 계수들에 의해 설정된 필터특성을 갖는 디지털 필터에서 k 비트(k는 4이상의 자연수)의 디지털 샘플들을 입력신호로서 수신하여 필터링하는 방법에 있어서:

상기 디지털 필터 계수들중의 임의의 계수들에 대한 패턴 중에서 비트이동, 비트가산, 또는 비트반전에 의해 정해진 공통패턴과 같아지는 패턴을 찾아 버츄얼 공통패턴을 형성한 후, 상기 디지털 샘플들을 상기 공통패턴 및 버츄얼 공통패턴에 대응되는 비트수 만큼 각기 쉬프팅하는 단계와;

상기 공통패턴에 대응되는 비트수 만큼 각기 쉬프팅된 상기 디지털 샘플들을 모두 합하여 공통계수 탭라인의 합성출력값을 구하는 단계와;

상기 공통계수 탭라인의 합성출력값을 공통입력으로 사용하여 상기 버츄얼 공통패턴에 대응되는 비트수 만큼 각기 쉬프팅된 상기 디지털 샘플들과 각기 합함에 의해 임의계수 탭라인들의 합성출력값을 각기 구하는 단계와;

상기 공통계수 탭라인의 합성출력값 및 상기 임의계수 탭라인들의 합성출력값을 누적적으로 합성하여 필터 출력값을 생성하기 위해 지연 및 가산을 순차로 행하는 단계를 가짐을 특징으로 하는 방법.

【청구항 8】

제7항에 있어서, 상기 디지털 필터가 이동통신 단말기의 중간주파수 단에 채용되는 경우에 상기 탭라인들의 수는 73으로 설정됨을 특징으로 하는 방법.

【청구항 9】

제8항에 있어서, 상기 CSD 코드워드의 비트수는 24비트임을 특징으로 하는 방법.

【청구항 10】

디지털 필터에 있어서:

k 비트의 디지털 샘플들을 입력신호로서 수신하여 CSD 코드내에서 n 비트의 코드워드로서 각기 표현되는 필터 계수들중에서 공통패턴으로 정의된 필터 계수의 비트 쉬프트 값들 만큼 각기 쉬프팅하는 제1쉬프트 레지스터 멤버들과, 상기 공통패턴으로 정의되지 않은 다른 필터 계수들의 코드워드를 상기 공통패턴의 코드워드를 이용하여 구현하기 위하여 쉬프팅하는 제2쉬프트 레지스터 멤버들을 포함하는 쉬프트 레지스터 그룹과;

상기 제1쉬프트 레지스터 멤버들로부터 출력된 쉬프팅된 디지털 샘플들을 모두 더하여 공통 탭라인에 제공하는 제1 합성멤버들과, 상기 제2쉬프트 레지스터 멤버들로부터

출력된 쉬프팅된 디지털 샘플들을 상기 공통 탭라인의 합성출력과 각기 더하여 각각의 대응되는 탭라인에 제공하는 제2 합성멤버들을 포함하는 덧셈그룹과;

상기 탭라인들에 연결되며 상기 합성출력들 간에 딜레이를 제공하기 위해 직렬로 연결된 복수의 지연기를 가지는 지연그룹과;

상기 지연기들의 출력과 상기 탭라인의 합성출력들을 더하여 k비트의 디지털 출력 신호를 생성하기 위한 복수의 덧셈기를 포함하는 출력 덧셈그룹을 적어도 포함함을 특징으로 하는 디지털 필터.

【청구항 11】

CSD 코드로 표현되는 필터 계수들을 가지는 N 탭 CSD 디지털 필터에 있어서:

복수 비트의 디지털 샘플들을 입력신호로서 수신하여 상기 필터 계수들중에서 공통패턴으로 정의되는 필터 계수의 비트 쉬프트 값들 만큼 각기 쉬프팅하는 제1쉬프트 레지스터 멤버들과, 상기 공통패턴으로 정의되지 않은 다른 필터 계수들의 코드워드를 상기 공통패턴의 코드워드를 이용하여 구현하기 위하여 쉬프팅하는 제2쉬프트 레지스터 멤버들을 포함하는 쉬프트 레지스터 그룹과;

상기 제1쉬프트 레지스터 멤버들로부터 출력된 쉬프팅된 디지털 샘플들을 모두 더하여 공통 탭라인에 제공하는 제1 덧셈기들과, 상기 제2쉬프트 레지스터 멤버들로부터 출력된 쉬프팅된 디지털 샘플들을 상기 공통 탭라인의 합성출력과 각기 더하여 각각의 대응되는 탭라인에 제공하는 제2 덧셈기들을 포함하는 덧셈그룹과;

상기 탭라인들에 연결되며 상기 합성출력들 간에 딜레이를 제공하기 위해 직렬로 연결된 복수의 지연기를 가지는 지연그룹과;

필터출력 응답을 제공하기 위해, 상기 지연기들의 출력과 상기 탭라인의 합성출력들을 더하여 k비트의 디지털 출력신호를 생성하기 위한 복수의 덧셈기를 포함하는 출력 덧셈그룹을 적어도 포함함을 특징으로 하는 디지털 필터.

【청구항 12】

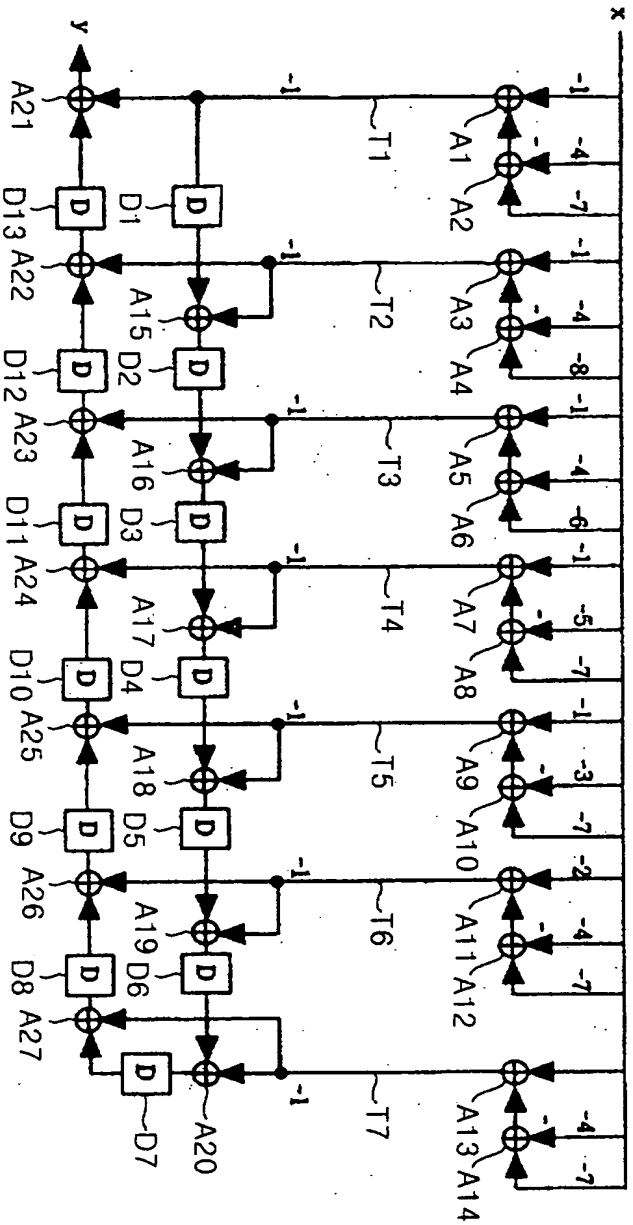
제11항에 있어서, 상기 공통 탭라인은, 상기 필터 계수들중 계수 패턴이 가장 공통성이 많은 필터 계수의 탭라인으로 설정됨을 특징으로 하는 디지털 필터.

【청구항 13】

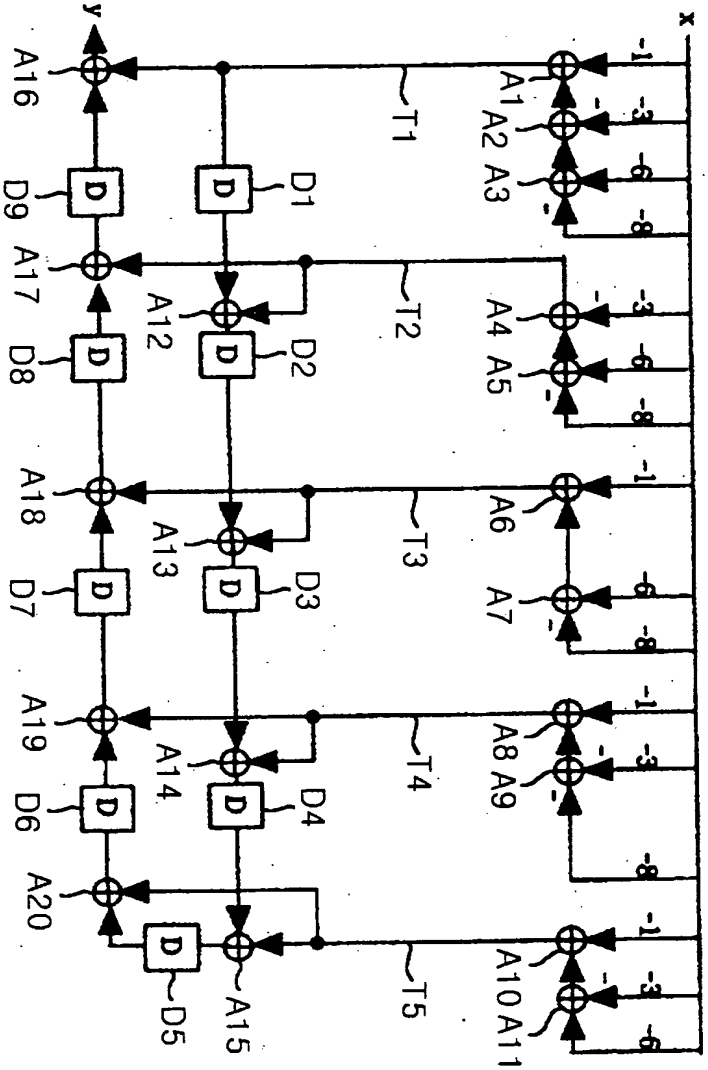
제11항에 있어서, 상기 디지털 필터는, 상기 쉬프팅, 덧셈, 지연을 행하는 디지털 신호처리 프로세서에 의해 소프트웨어적으로 구현됨을 특징으로 하는 디지털 필터.

【도면】

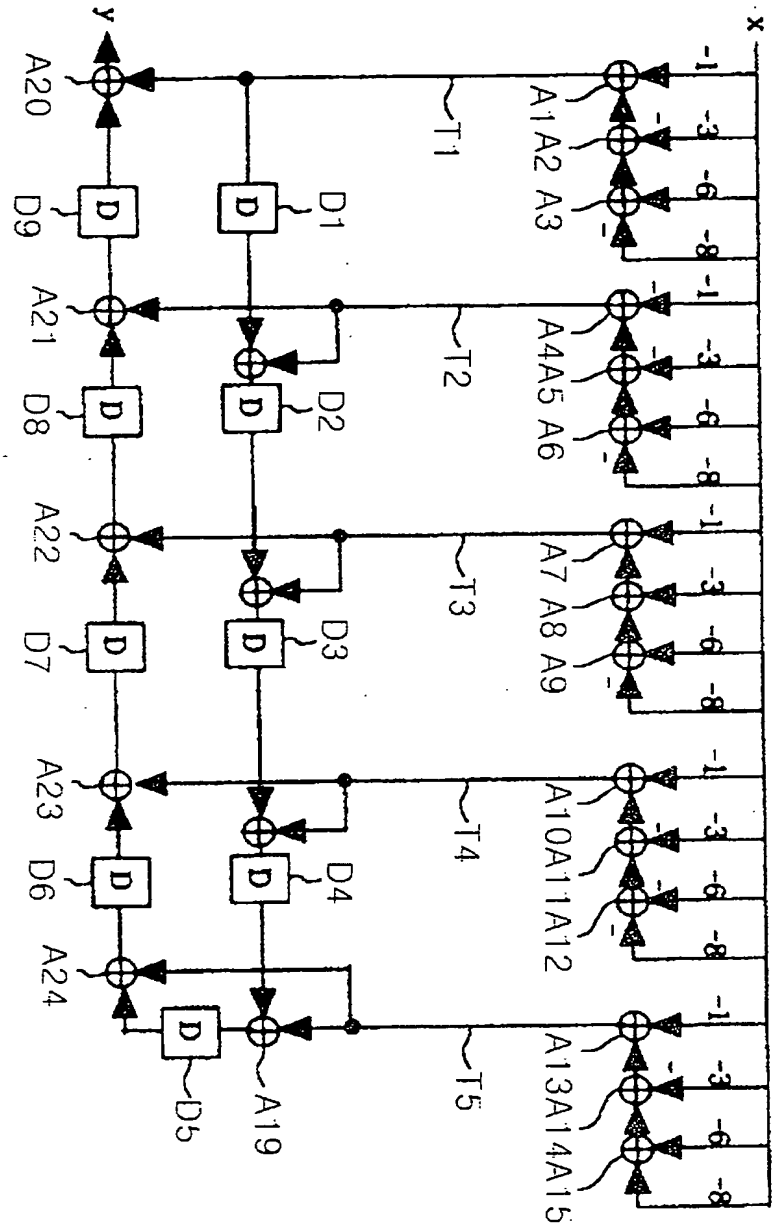
【도 1】


x
y
A21
A22
A23
A24
A25
A26
A27
A20
D13
D12
D11
D10
D9
D8
D7
D1
D2
D3
D4
D5
D6
A15
A16
A17
A18
A19
T1
T2
T3
T4
T5
T6
T7
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14

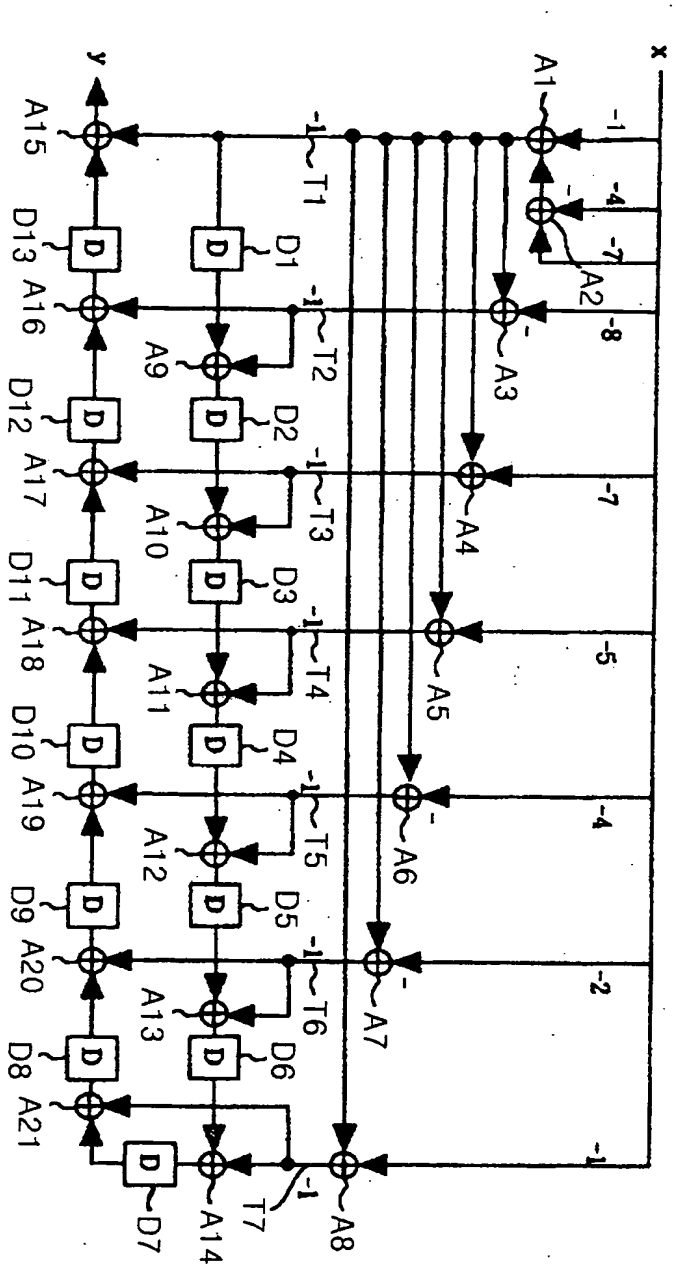
【도 2】


x
y
-1
-3
-6
-8
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
A16
A17
A18
A19
A20
D
D1
D2
D3
D4
D5
D6
D7
D8
D9
T1
T2
T3
T4
T5

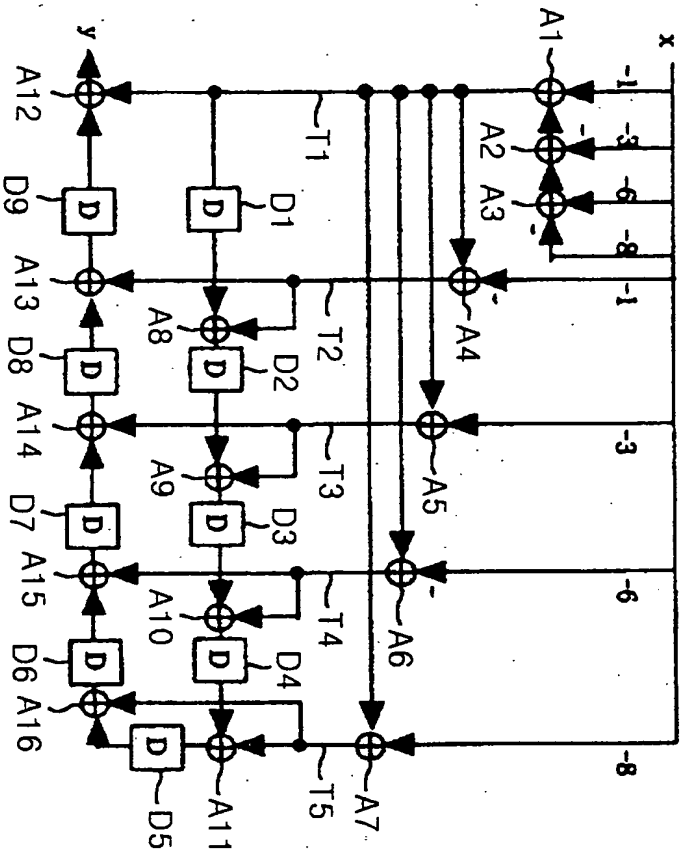
【도 3】


x
y
-1
-3
-6
-8
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
A19
A20
A21
A22
A23
A24
T1
T2
T3
T4
T5
D
D1
D2
D3
D4
D5
D6
D7
D8
D9

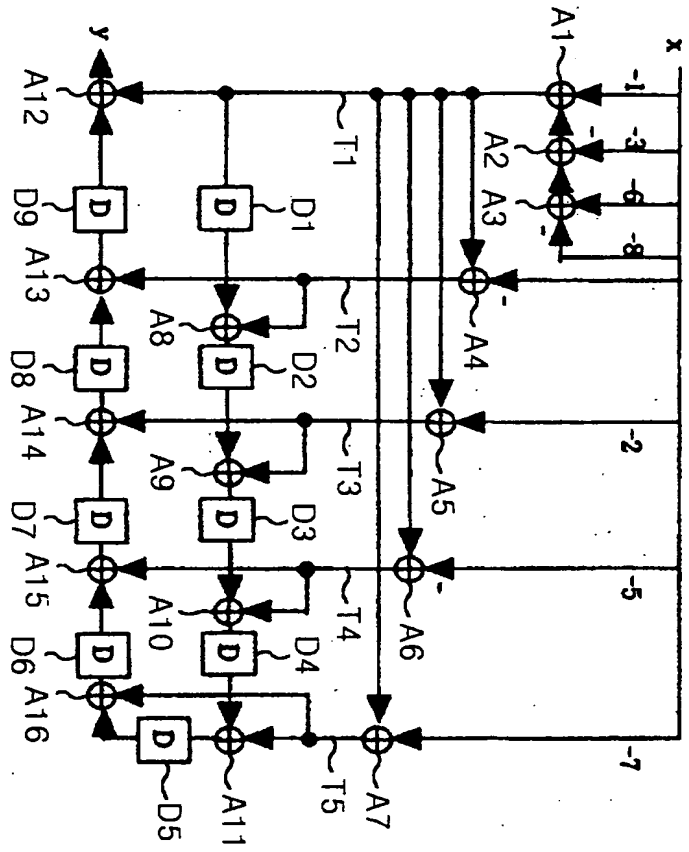
【도 4】


x
y
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
A16
A17
A18
A19
A20
A21
T1
T2
T3
T4
T5
T6
T7
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
D
-1
-4
-7
-8
-7
-5
-4
-2
-1

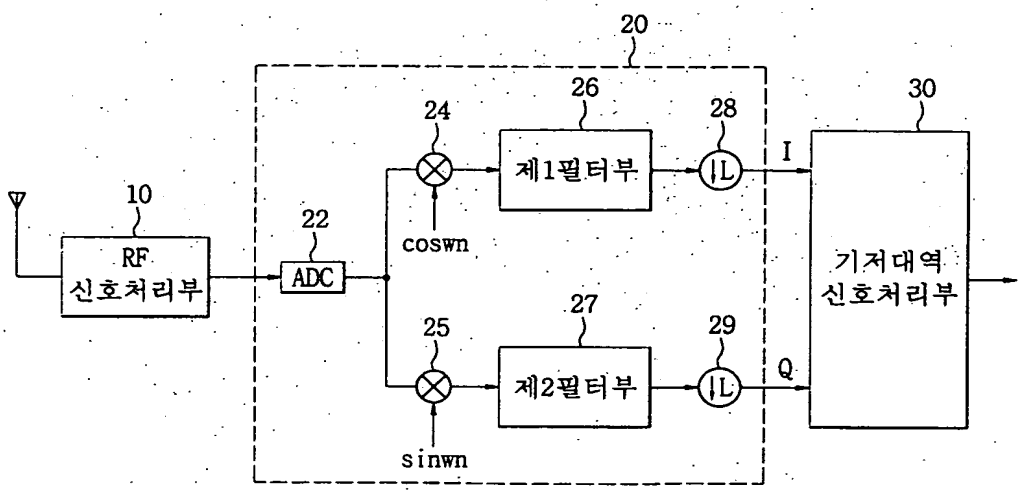
【도 5】


x
y
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
A16
D1
D2
D3
D4
D5
D6
D7
D8
D9
T1
T2
T3
T4
T5
-1
-3
-6
-8
D

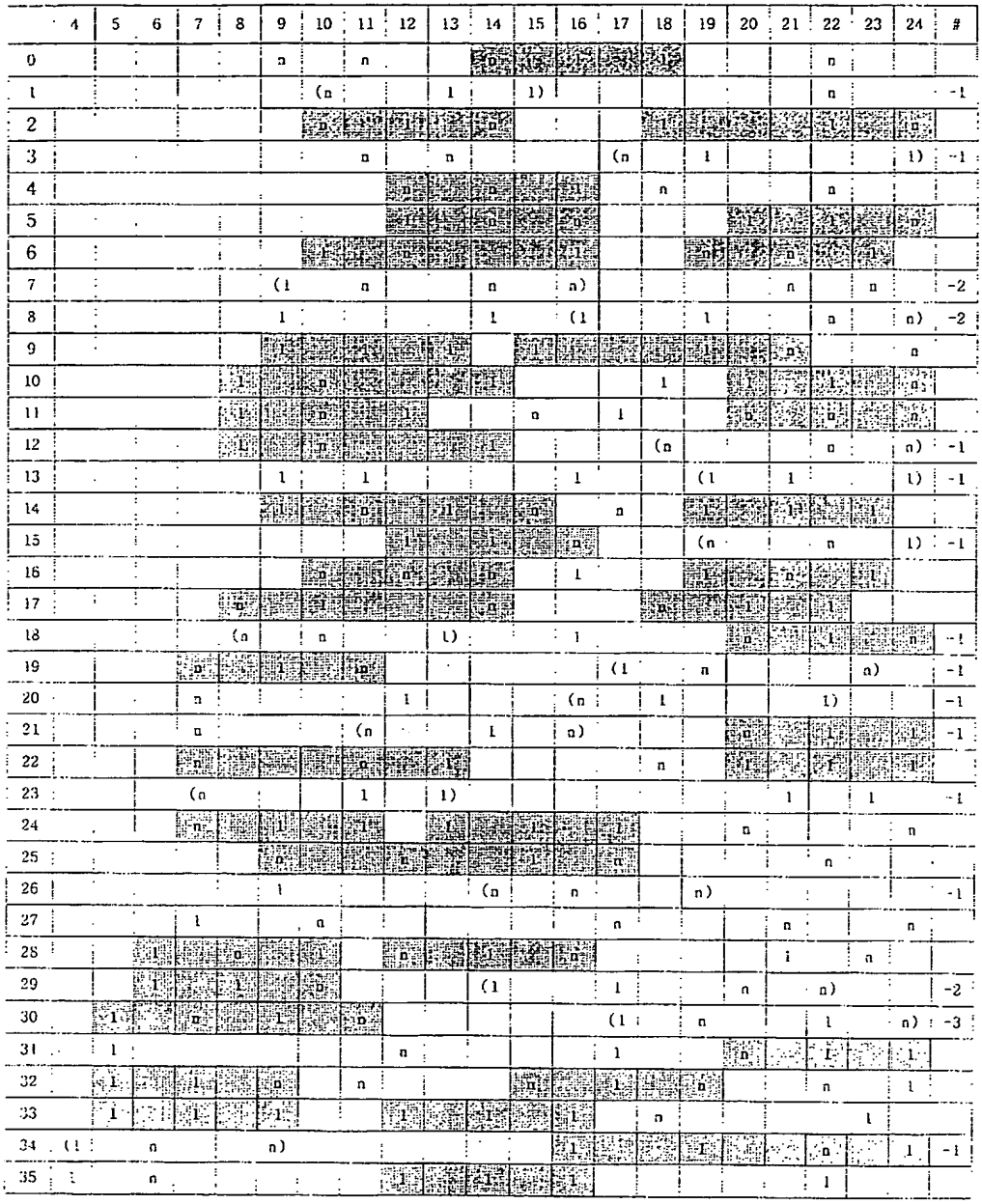
【도 6】


x
y
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
A16
D1
D2
D3
D4
D5
D6
D7
D8
D9
T1
T2
T3
T4
T5
-1
-3
-6
-8
-2
-5
-7

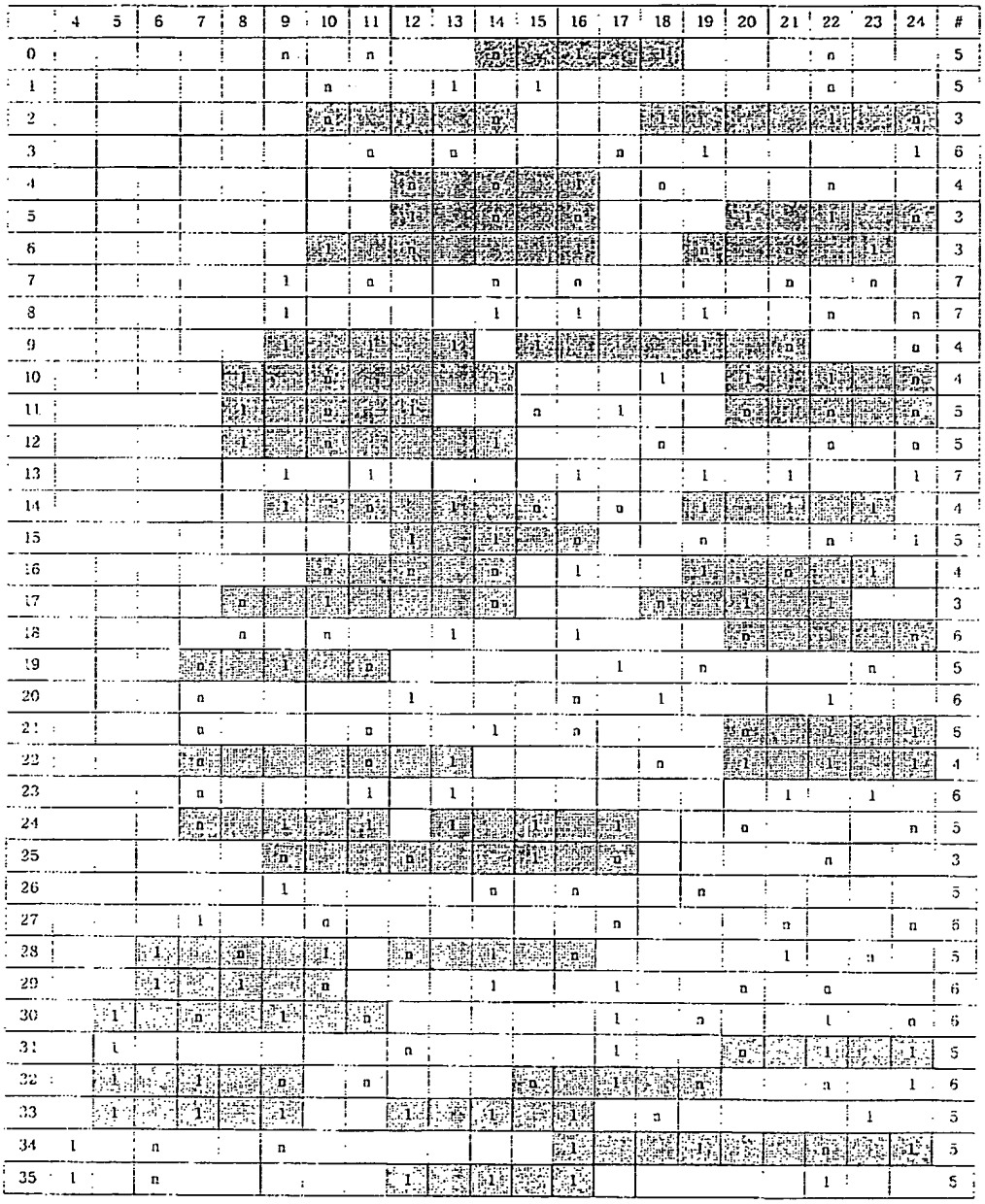
【도 7】


20
10
RF
신호처리부
22
ADC
24
coswn
25
sinwn
26
제1필터부
27
제2필터부
28
29
IL
IL
I
Q
30
기저대역
신호처리부

【도 8】


4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 #
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35

【도 9】


4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 #
0 5
1 5
2 3
3 6
4 4
5 3
6 3
7 7
8 7
9 4
10 4
11 5
12 5
13 7
14 4
15 5
16 4
17 3
18 6
19 5
20 6
21 5
22 4
23 6
24 5
25 3
26 5
27 6
28 5
29 6
30 6
31 5
32 6
33 5
34 5
35 5